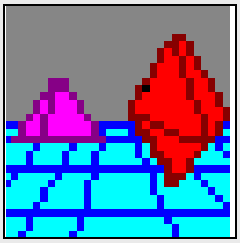
MAPPER



Version 1.4.08
Mars 2006

Manuel Utilisateur

TX2
Produits et Services EDI

TX2 CONCEPT

Table des matières

1. Généralités
2. Organisation des fichiers
3. Créer un format interne
4. Les enregistrements
5. Utilisation des structures
6. Structure d'un enregistrement
7. Les variables
7.1 Utilisation des variables dans un format Applicatif vers EDI
7.2 Utilisation des variables dans un format EDI vers Applicatif
8. Constantes et expressions
8.1 Constantes
8.2 L'opérateur +
8.3 Les variables
8.4 Les fonctions pré-définies
8.5 La structure "SWITCH" (instructions conditionnelles)
9. Sauvegarder
10. Compiler
11. Tables de transcodifications
12. Particularités des formats EDI-EDI
13. Particularités du standard GENCOD
13.1 Gestion des rubriques 177
13.2 Segments de service
14. Particularités du standard TRADACOM
15. Particularités du standard VDA
Architecture de l'environnement de développement TX2
Annexe : Gestion des tables de références INHREF et MESSREF

1. GENERALITES.

Le gestionnaire de formats internes, ou mapper, permet de paramètrer des liens fichiers applicatifs <-> messages EDI. Il permet de paramètrer à la fois la structure physique du fichier applicatif, ainsi que les liens avec les données du message EDI.

Les fichiers applicatifs sont des fichiers ASCII, multi ou mono - enregistrements. Voici un exemple de fichier applicatif, et le message EDI ORDERS correspondant :

Fichier applicatif :

```
001HBTECHNOLOGIES ADRDEST        HB TECHNOLOGIES     DICROP
12
002TX2 TRADUCTEUR EDI     2   PCE
002GESTIONNAIRE DE FORMAT 1   PCE
001HBTECHNOLOGIES ADRDEST        HB TECHNOLOGIES     DICROP
12
002TX2 TRADUCTEUR EDI     2   PCE
002GESTIONNAIRE DE FORMAT 1   PCE
```

Fichier EDIFACT :

```
UNB+UNOA:1+HB TECHNOLOGIES+ADRDEST+951230:1543+I0000861'
UNH+268+ORDERS:D:93A:UN'
BGM+12'
NAD+SU+HB TECHNOLOGIES'
NAD+BY+DICROP'
UNS+D'
LIN++TRADUCTEUR EDI'
QTY+:2+PCE'
LIN++GESTIONNAIRE DE FORMAT'
QTY+:1+PCE'
UNT+10+268'
UNH+269+ORDERS:D:93A:UN'
BGM+12'
NAD+SU+HB TECHNOLOGIES'
NAD+BY+DICROP'
UNS+D'
LIN++TRADUCTEUR EDI'
QTY+:2+PCE'
LIN++GESTIONNAIRE DE FORMAT'
QTY+:1+PCE'
UNT+10+269'
UNZ+2+I0000861'
```

On constate que :

1. le fichier applicatif contient 2 types d'enregistrements, repérés par des **étiquettes d'enregistrement** 001 et 002. Le choix des étiquettes est complètement libre.
2. Des données ou des qualifiants ont été automatiquement générées.
3. Lorsque le fichier contient plusieurs messages, les messages doivent se suivre.

2. ORGANISATION DES FICHIERS DE PARAMETRAGE

Un format applicatif (appelé aussi format interne) est identifié par un code que vous lui attribuez lors de la création.

Par exemple : **COMMANDE.**

Ce format fait référence à un message ou subset et un seul.
Par exemple : **ORDER93A**.

Les fichiers de paramètrage générés seront alors les suivants :

Paramètrages sources : **\TX2\DATA\COMMANDE.REW**, et **\TX2\DATA\COMMANDE.SEQ** qui produiront eux-mêmes automatiquement le fichier **\TX2\DATA\COMMANDE.DEF** lors de la compilation.

Paramètrage compilé (exploité par le traducteur TX2) :
\TX2\COMMANDE.FDX.

Par ailleurs, le format interne sera référencé dans la table de référence **\TX2\INHREF.RDX**, dans laquelle sont stockées les propriétés du format interne ainsi que la correspondance Format <-> Message. Cette table peut être visualisée et maintenue "manuellement", en utilisant l'utilitaire **BPT.EXE** décrit en annexe.

Règles de gestion du traducteur :

En émission, le format à utiliser pour traduire un fichier devra être précisé à l'appel du traducteur, soit explicitement, soit en nommant le fichier à traduire **<Code format>.INH**.

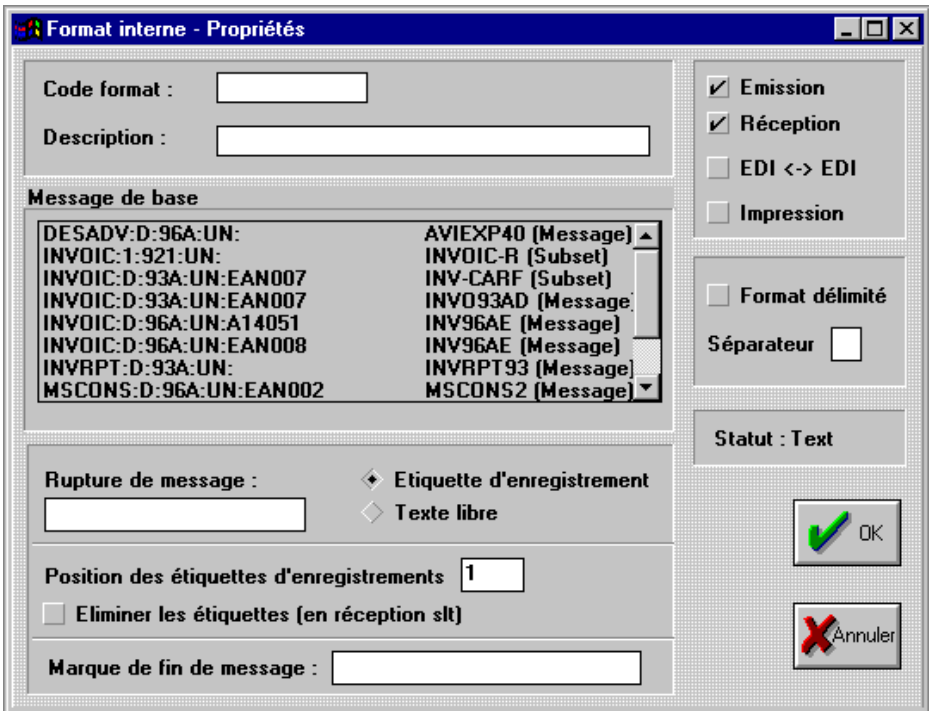
En réception, le traducteur recherche dans la table **MESSREF.RDX** le code message correspondant à ce qui est indiqué dans le segment **UNH** (par exemple **ORDERS D 93A UN** donne le code **ORDER93A**), puis le traducteur recherche dans la table **\TX2\INHREF.RDX** le code format correspondant à ce code message (par exemple **ORDER93A** donne **COMMANDE**).

Note : on peut faire dépendre le format interne du partenaire. Ceci se fait en utilisant la gestion des applications et des autorisations par message de la station TX2.

Si vous utilisez la station TX2, les fichiers en format applicatif à traduire doivent s'appeler <Code format>.INH dans le répertoire \WSTA, et les fichiers en format applicatif reçus s'appelleront <Code format>.RCV, dans le répertoire \WSTA. ***Mais ces contraintes peuvent être évitées en utilisant des paramètres de traduction personnalisés. Ces paramètres sont décrits dans la documentation "TX2 Traducteur - Documentation technique run-time".***

3. CREER UN FORMAT INTERNE.

Pour créer un format interne, sélectionnez l'option **Fichier - Nouveau format**. Le dialogue suivant apparaît :



Vous devez saisir les propriétés du format interne :

- Code format interne : c'est un nom que vous donnez au format à créer. Lorsque le traducteur est utilisé au sein de **TX2 Station**, le code format conditionne le nom du fichier à traduire en émission (**`<Code format>.INH`**) et le nom du fichier traduit en réception (**`<Code format>.RCV`**).

- Description : description succincte du format.

- Message de base : vous devez sélectionner dans la liste le message ou le subset auquel ce format interne se rapporte.

- Emission, Réception, Impression ,EDI->EDI : c'est la direction pour laquelle ce format sera utilisé. Emission signifie que ce format sera utilisé pour aller du format applicatif vers le format EDI, Réception signifie que ce format sera utilisé pour aller du format EDI vers le format applicatif, et Impression signifie que ce format sera utilisé pour obtenir une sortie imprimable du message EDI. *En dehors du cas des formats "pivots standards" (en particulier les formats générés automatiquement par l'atelier de conception), il est conseillé de construire 2 formats différents pour l'émission et la réception, car certaines instructions avancées ont des comportements spécifiques au sens du format interne*. Le type **EDI->EDI** permet de construire des formats de traduction permettant de traduire directement un document EDI standard vers un autre document EDI standard, par exemple une facture INVOIC 90.1 en INVOIC D93A. Dans ce cas, le format interne doit être construit à partir de la base générée automatiquement par l'atelier de conception (Option Générer format EDI -> EDI). Un format EDI->EDI est traité comme un format en émission (en particulier les règles de contrôles de présence des données s'appliquent).

- Rupture de message : c'est le critère qui permet au traducteur de savoir quand commence un nouveau message dans le format applicatif. Par exemple, dans le format du paragraphe 1, c'est **001**.

- Etiquette d'enregistrement, Texte libre : ce menu permet d'indiquer si le critère de rupture de message décrit plus haut est une étiquette d'enregistrement (comme c'est le cas dans l'exemple du paragraphe 1), ou si c'est une chaîne de caractères placée là uniquement pour ça. Par exemple si la structure du format interne est :

```
ENTETECDE01   19980101...
LIGNE ARTICLE1   3000FRF
REMISE10.5
```

```
REMISE05.5
LIGNE ARTICLE2    450FRF
REMISE10.5
ENTETECDE02   19980101...
LIGNE ARTICLE1   3000FRF
REMISE10.5
 LIGNE ARTICLE2    450FRF
REMISE10.5
...
```

le critère de rupture de message sera **ENTETE**, et c'est une étiquette d'enregistrement.

Note : en émission, **TX2** a toujours besoin d'étiquettes pour identifier les enregistrements. Dans le cas d'un fichier mono-enregistrement sans étiquette, le fichier doit être "pré-traité" à l'aide de l'utilitaire **UTILTX2.EXE** (consulter le support technique H.B. Technologies).

```
E;NOCDE;;HBTECHNOLOGIES;;;1
```

- Marque de fin de message : permet de spécifier une expression qui sera systèmatiquement écrite en fin de chaque document dans le fichier en format interne. Par exemple si on souhaite que chaque document soit cloturé par la chaîne de caractère *****FINMESS**, l'expression saisie doit être :

Si l'expression de marque de fin de message est du type :

$INCLUDE(Fichier), alors les instructions contenues dans **Fichier** sont insérées dans un bloc LASTDATASET, qui correspond à une suite d'instructions exécutées en fin de chaque message.

Exemple : `$INCLUDE (INVOIC93.INC).`

Note importante : *Cette instruction a été maintenue par souci de compatibilité ascendante. Il est préférable de définir les instructions de fin de message l'enregistrement de nom d'étiquette réservée* ***$-END****.*

Une fois ces informations saisies, deux fenêtres s'ouvrent : une fenêtre contenant le diagramme de branchement du message EDI, et une fenêtre qui contiendra la liste des enregistrements du format interne. Au départ, seule une ligne Déclarations apparaît. Il s'agit d'un enregistrement particulier, dans lequel on déclarera soit des données auto-générées de niveau message, soit des variables tampons.
(Note : la partie Déclarations correspond à un enregistrement particulier dont l'étiquette interne est GENINH0002).

Contrôle des données obligatoires :

Lorsque vous définissez un format dans le sens **Emission** ou **Emission-réception**, vous devez référencer dans ce format interne toutes les données obligatoires du message. Lors de la compilation, TX2 vérifie que votre format est cohérent. En particulier, si vous référencez une donnée d'un segment non obligatoire, alors seulement TX2 vérifie que les données obligatoires de ce segment sont référencées.

Quoi qu'il en soit, dans un format Emission ou Emission-Réception, vous devez référencer les données de service suivantes :

Adresse émetteur	**UNB 0004**
Adresse destinataire	**UNB 0010**
Date de l'échange	**UNB 0017**
Heure de l'échange	**UNB 0019**
N° de message	**UNH 0062**

Nous vous conseillons de fournir les données **UNB 0017, UNB 0019** et **UNB 0062** dans la partie déclaration sous forme de données auto-générées : **$DATE, $TIME** et **$2+=1**. (pour plus de détails sur la génération de constantes ou expressions, voir le paragraphe 7).

Les autres données de service obligatoires sont fournies par le système.

***Remarques** :*
En émission, les données UNB.S001.0001 (identifiant de syntaxe) et UNB.S001.0002 (numéro de version de syntaxe) ne doivent pas être mappées, car elles sont alimentées en fonction de paramètres de traduction ou en fonction du destinataire du message. Les données du composite UNB.S009 sont également automatiquement alimentées en émission.
La donnée UNB.0020 (numéro de contrôle d'échange) est alimentée par un compteur interne par défaut. Il est possible de forcer ce compteur en mappant la donnée dans le format interne.

Pour un format défini en réception uniquement, aucun contrôle de présence de donnée n'est effectué.

4. LES ENREGISTREMENTS.

La fenêtre principale du format interne (celle dont le titre est le code format interne) contient la liste des enregistrements déclarés pour ce format interne, plus la partie Déclarations. Il peut également apparaître des informations concernant la structure globale du fichier (répétitions, indentations, existence d'entités **STRUC** et **SEQ STRUC**) : ces notions sont détaillées dans le ***chapitre 5 - Utilisation des structures***.

- Créer un enregistrement (bouton Ins ou menu Enregistrements-Nouveau) : Dans ce cas, vous devez indiquer l'étiquette de l'enregistrement (par exemple 002) Vous pouvez également indiquer un libellé explicatif, ainsi qu'un numéro de séquence (voir page suivante). La fenêtre de l'enregistrement s'ouvre alors.
- Détruire un enregistrement (bouton ciseaux ou menu Enregistrements-Détruire)
- Renommer un enregistrement : cela consiste à attribuer une autre étiquette.
- Dupliquer un enregistrement.
- Editer et modifier la structure de l'enregistrement (Double click).

Note sur les numéros de séquence :

Il est également possible de forcer un *numéro de séquence* pour l'enregistrement.

Par défaut, le système stocke les enregistrements dans l'ordre où ils ont été créés. Ceci ne pose pas de problème, car le traducteur trie le format interne généré en fonction de l'ordre des éléments de données mappés dans le message.

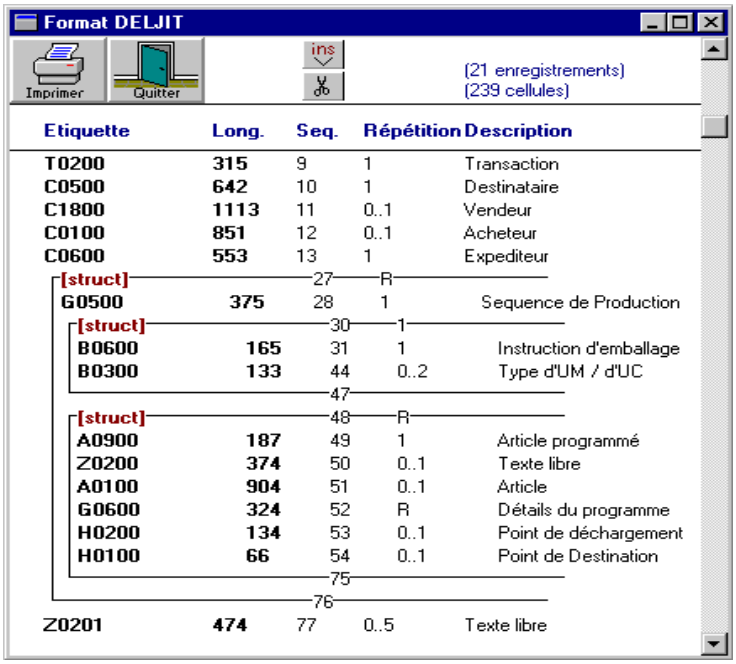
Cependant, il est des cas où l'ordre de déclaration des enregistrements est important, par exemple lorsqu'il est fait référence à une variable qui se trouve déclarée dans un autre enregistrement. Dans ce cas, l'enregistrement où est déclarée la variable doit forcément être déclaré avant l'enregistrement dans lequel cette variable est utilisée.

L'attribution des numéros de séquence est complètement libre, et aucun contrôle n'est effectué par le système.

5. UTILISATION DES STRUCTURES.

De manière générale , lorsqu'on décrit un format interne, il n'est pas nécessaire d'indiquer explicitement au système l'ordre dans lequel les enregistrements apparaissent dans le fichier. Cet ordre est la plupart du temps « implicitement logique », c'est-à-dire qu'il est conditionné par l'ordre des informations dans le message EDI.

Cependant, dans certains cas, l'ordre et la structure du fichier interne doivent être explicitement indiqués. L'écran suivant montre un exemple de format interne dans lequel les indications de structure ont été saisies :


Format DELJIT
Imprimer
Quitter
ins
(21 enregistrements)
(239 cellules)
Etiquette Long. Seq. Répétition Description
T0200 315 9 1 Transaction
C0500 642 10 1 Destinataire
C1800 1113 11 0..1 Vendeur
C0100 851 12 0..1 Acheteur
C0600 553 13 1 Expediteur
[struct] 27 R
G0500 375 28 1 Sequence de Production
[struct] 30 1
B0600 165 31 1 Instruction d'emballage
B0300 133 44 0..2 Type d'UM / d'UC
47
[struct] 48 R
A0900 187 49 1 Article programmé
Z0200 374 50 0..1 Texte libre
A0100 904 51 0..1 Article
G0600 324 52 R Détails du programme
H0200 134 53 0..1 Point de déchargement
H0100 66 54 0..1 Point de Destination
75
76
Z0201 474 77 0..5 Texte libre

Note sur les formats d'impression :

Pour un format d'impression, le bouton ou l'option de menu **Fichier - En-tête / Pied de page** permettent d'accéder à un écran de saisie des expressions en-tête et pied de page.

Les indications de structures permettent de forcer l'ordre et les niveaux de répétitions des enregistrements. Ceci doit être utilisé lorsque la structure du fichier interne est éloignée de la structure du message, afin de lever toute ambiguïté sur le traitement du message.

Pour définir une structure, il faut sélectionner un groupe d'enregistrements (en maintenant le bouton gauche de la souris appuyé), et sélectionner l'option Définir STRUCTure ou Définir SEQ STRUCture du menu Enregistrements.

La notion de structure implique de renseigner les facteurs de répétition des différents enregistrements (Option **Enregistrements - Répétition**).

Différence entre STRUCT et SEQ STRUCT :

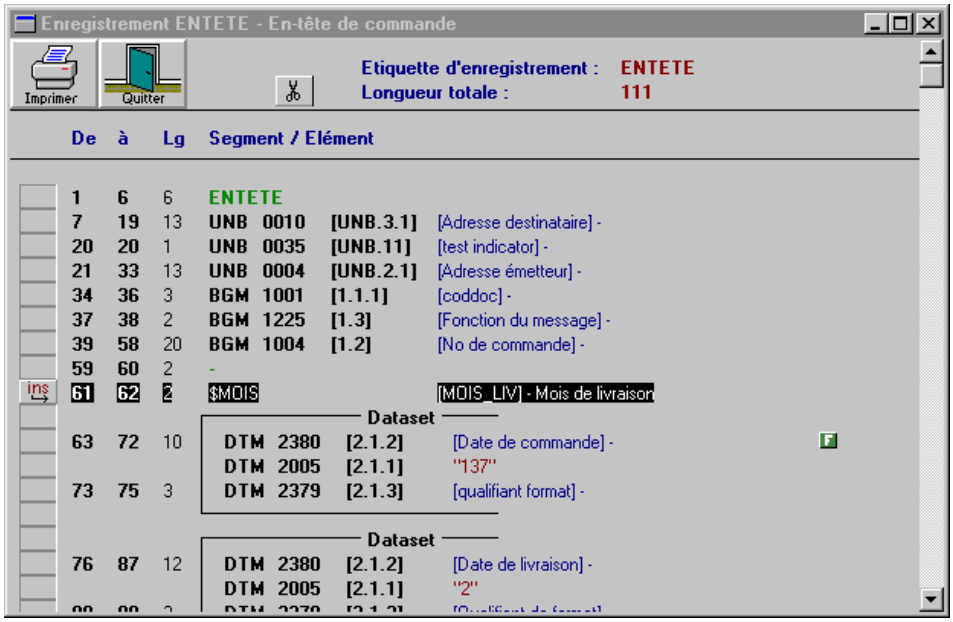
La différence entre les deux se manifeste en traduction de EDI vers format interne.

Le système parcoure une STRUCT dans l'ordre où sont déclarés les enregistrements, le premier enregistrement de la STRUCT étant un enregistrement Trigger qui conditionne le balayage du reste de la STRUCT. Dans le cas d'une SEQ STRUCT, celle-ci est parcourue en boucle indéfiniment tant qu'il y a des enregistrements à alimenter.

De manière générale, on utilise les STRUCT.

6. STRUCTURE D'UN ENREGISTREMENT.

Voici un exemple de fenêtre de structure d'enregistrement :


Enregistrement ENTETE - En-tête de commande
Imprimer
Quitter
Etiquette d'enregistrement : ENTETE
Longueur totale : 111
De à Lg Segment / Elément
1 6 6 ENTETE
7 19 13 UNB 0010 [UNB.3.1] [Adresse destinataire] -
20 20 1 UNB 0035 [UNB.11] [test indicator] -
21 33 13 UNB 0004 [UNB.2.1] [Adresse émetteur] -
34 36 3 BGM 1001 [1.1.1] [coddoc] -
37 38 2 BGM 1225 [1.3] [Fonction du message] -
39 58 20 BGM 1004 [1.2] [No de commande] -
59 60 2 -
ins 61 62 2 $MOIS [MOIS_LIV] - Mois de livraison
Dataset
63 72 10 DTM 2380 [2.1.2] [Date de commande] -
DTM 2005 [2.1.1] "137"
73 75 3 DTM 2379 [2.1.3] [qualifiant format] -
Dataset
76 87 12 DTM 2380 [2.1.2] [Date de livraison] -
DTM 2005 [2.1.1] "2"

La ligne courante apparaît sur fond noir. Pour sélectionner un bloc de cellules, cliquer sur la première cellule, puis pointer sur la dernière cellule du bloc en cliquant tout en maintenant la touche Shift (ou Majuscule) appuyée.

Une définition d'enregistrement est constituée de cellules, dont le type peut-être :

- Mapping : il s'agit d'une cellule permettant de mettre en correspondance une donnée du fichier interne avec une donnée du message EDI (par exemple BGM.1001 dans l'exemple ci-dessus).

Les champs à saisir sont :

- Segment : c'est le segment à mettre en correspondance.

- Groupe : si le segment apparaît plusieurs fois dans le message, il faut préciser le numéro de groupe.

- Elément : choisir l'élément de donnée à mettre en correspondance.

- Longueur : longueur dans le fichier interne (indiquer 0 pour une déclaration de variable).

- Cible : Cocher si c'est une donnée cible, c'est à dire si vous souhaitez répéter la valeur courante d'une donnée déjà mappée par ailleurs.

- Obligatoire : Cocher si la donnée doit obligatoirement être présente dans le fichier en fomat interne. Ceci permet d'introduire des contrôles de présence supplémentaires par rapport à la norme.

- Référence : zone libre dans laquelle vous pouvez une référence applicative.

- Libellé : zone libre dans laquelle vous pouvez indiquer un commentaire.

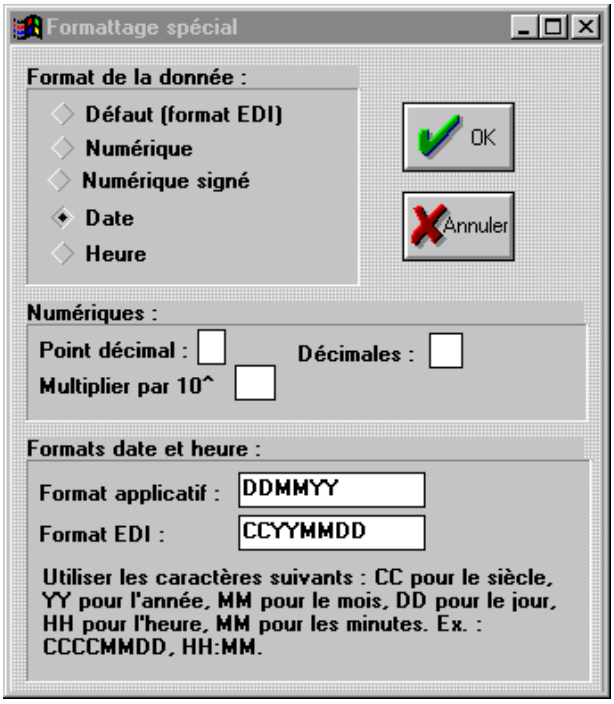
- Stocker dans : Indiquer un nom de variable dans laquelle vous souhaitez stocker la valeur courante de l'élément de donnée.

- Bouton **Transco :** Permet d'accéder au dialogue de saisie afin d'indiquer si une transcodification doit s'opérer. Dans ce cas, vous devrez saisir sélectionner une table de codes qui aura préalablement été définie (voir paragraphe 8).

- Bouton **Spécial :** Permet de préciser le type de la donnée et éventuellement des conversions de format. Par défaut, le type de la donnée est le type EDI

En particulier, ce bouton permet d'effectuer des conversions de format de date, tout en offrant le support de l'an 2000.

Voici un exemple d'écran "*Spécial*" :



Pour les données de type Date, le traducteur effectue une conversion de format entre le format applicatif et le format EDI, en s'appuyant sur les correspondances de caractères d'une zône à l'autre.
Dans le cas présent, une date 241098 sera transcodée en 19981098. Le système rajoute automatiquement le siècle en considèrant que les années postérieures à 50 sont 19xx, sinon on choisit 20xx.

Autres exemples :

YYMMDD	**DD/MM/YY**
HHMM	**HH:MM.**

Voir également la fonction $CC, paragraphe 7.

Note sur les formats numériques : dans le cas d'un format numérique, le champ ***multiplier par*** permet d'effectuer un décalage de décimale. Par exemple si une donnée de format interne est fournie en Grammes et qu'on souhaite l'émettre en KG, alors il faudra diviser la valeur par 1000, donc spécifier **-3** dans le champ *Multiplier par* (c'est à dire on multiplie la valeur de la donnée par 10^{-3}).

- Bouton **Condition :** Permet d'indiquer une contrainte conditionnelle. Par exemple, le qualifiant de l'adresse doit valoir "SU". La contrainte SEQ_WITH (case à cocher) doit être utilisée lorsqu'on souhaite mettre une condition sur un segment répétable indéfiniment (comme le segment LIN).
- *Auto-générée :* il s'agit d'une cellule permettant d'alimenter automatiquement une donnée du message EDI (donc valable uniquement dans le sens format interne vers EDI). Les données à saisir sont le segment, groupe et élément (Cf. mapping) ainsi que la valeur à générer, qui est une expression pouvant combiner des chaînes de caractères (entre "") et les fonctions $DATE (date du jour), $TIME (heure courante), $i+= (incrémentation de compteur) et $PAD (paddage de donnée).

Exemple :

`"XYZ" + $DATE + "X" + $TIME + "X" + $PAD (4,L,0,$4+=1)` donne :

```
XYZ960123X1523X0002
```

(pour plus de détails sur la génération de constantes ou expressions, voir le paragraphe 7)

- Filler : il s'agit d'une cellule permettant de déclarer un Filler dans l'enregistrement. Cette cellule est appelée SKIP. Vous devez saisir la longueur du filler.

- Texte : il s'agit d'une cellule permettant de forcer un texte dans le fichier applicatif (donc valable uniquement dans le sens Réception). Ce texte est une expression, qui suit les mêmes règles que pour les données auto-générées. Vous devez indiquer la longueur du texte.

- Calcul : il s'agit d'une cellule d'affecter à une variable le résultat d'une expression.

- Texte libre : il s'agit d'un bloc d'instructions qui sera inséré dans le fichier source de définition de format. Il permet d'utiliser des instructions du traducteur non supportées par le mapper.

- Variable : il s'agit d'une cellule permettant de mettre en correspondance une donnée du fichier interne avec une variable qui aura été déclarée au préalable. Ces variables sont déclarées en indiquant un identifiant dans le champ Stocker dans du dialogue de saisie d'un mapping.

Pour plus de précisions concernant l'utilisation de variables, veuillez vous reporter au paragraphe suivant (***6 - Les variables***).

Vous devez saisir ou sélectionner un nom de variable, la longueur et éventuellement une référence interne. De plus vous pouvez préciser une instruction de transcodification (bouton transco) ou une règle de formatage (bouton Spécial).

- Dataset : un dataset est une sous-structure à l'intérieur d'un enregistrement. Un Dataset doit être utilisé lorsque on souhaite avoir 2 occurrences d'un segment ou groupe de segments à l'intérieur d'un même enregistrement, sans que toutes les données ne soient qualifiées.

Pour définir un dataset, vous devez sélectionner un ensemble de cellules (click gauche puis Shift-click gauche), puis sélectionner l'option de menu *Données - Définir Dataset*.

7. LES VARIABLES.

Vous avez la possibilité de définir des variables tampons dans une description de format interne. Ces variables tampon servent à stocker des valeurs trouvées soit dans le fichier EDI, soit dans le fichier interne, et de les utiliser plus tard, ce qui permet de contrôler plus facilement les instructions de mapping, en particulier lorsqu'il est nécessaire de ré-ordonner les données.

7.1. Utilisation des variables dans un format Applicatif vers EDI

Dans un format émission, une variable sert à stocker un champ d'un enregistrement du fichier interne. Pour ce faire, on insère une cellule de type Variable, à laquelle on attribue un nom et dont on précise la longueur et la position dans le fichier applicatif. Par exemple, variable NOMDEST, position 5, longueur 20.

La valeur stockée dans cette variable peut alors être réutilisée dans une cellule de type Auto-générée, soit en prenant la valeur complète, soit en en prenant une sous-chaîne. Cette valeur peut d'ailleurs être combinant avec d'autres variables ou fonctions TX2.

Par exemple, si on doit mettre dans l'élément NAD 3039 les 6 premiers caractères de NOMDEST, préfixés par "301", on insérera une donnée auto-générée mappant vers l'élément NAD 3039 et dont la valeur sera :
`"301" + $SUB (0,6,$NOMDEST)`.

On notera au passage 2 éléments importants :

Lorsqu'on référence une variable dans une expression, on doit préfixer son nom par le signe $.

La fonction $SUB (sous-chaîne) prend comme arguments : l'offset départ (commençant à 0), la longueur, et l'expression source.

Autre exemple d'utilisation :

Dans un enregistrement, la date d'expédition est en 2 parties : de la position 7 à 10, les mois et jour (MMJJ) et de la position 15 à 16, l'année (AA), alors que la date à générer doit être sous la forme AAAAMMJJ.
Il suffit de déclarer la variable MOISJOUR en position 7 et la variable ANNEE en position 15, puis de déclarer une donnée auto-générée mappant vers le DTM et ayant pour valeur :

```
$SUB (0,2,$DATE2) + $ANNEE + $MOISJOUR.
```

7.2. Utilisation des variables dans un format EDI vers Applicatif

Dans un format réception, une variable sert à stocker un élément de donnée EDI. Pour ce faire, on insère une cellule de type Mapping, et on renseigne le champ de saisie ***Stocker Dans***. La cellule mapping peut être déclarée à la volée dans un Enregistrement, ou peut-être déclarée dans la partie ***Déclarations***, et peut avoir une longueur nulle ou non. (Dans la partie Déclarations, elle doit avoir une longueur nulle).

Pour utiliser la valeur stockée dans une variable, il suffit d'insérer une cellule ***Variable***. Un dialogue de saisie apparaît, dans lequel vous devez saisir ou piocher dans la liste un nom de variable, la longueur dans l'enregistrement, et éventuellement une instruction de transcodification ou une instruction de formatage spécial.

Exemple d'utilisation :

Dans le format interne, on souhaite répéter dans tous les enregistrements la valeur de la donnée NAD 3039 SU à 2 endroits différents, avec 2 tables de transcodification différentes. Pour ce faire, on déclare dans la partie Déclaration un mapping vers NAD 3039 avec une condition sur le qualifiant, en précisant COD-DEST dans le champ Stocker dans.

Puis dans l'enregistrement, on insère aux endroits adéquats 2 cellules ***Variable***, en appliquant à chaque fois la transcodification.

L'utilisation d'une partie Déclaration dans laquelle on référence les données permet d'obtenir un script de format interne bien structuré et lisible.

Autre exemple d'utilisation :

Dans un enregistrement, la date d'expédition est en 2 parties : de la position 7 à 10, les mois et jour (MMJJ) et de la position 15 à 16, l'année (AA), alors que la date à générer doit être sous la forme AAAAMMJJ.
Il suffit de déclarer la variable MOISJOUR en position 7 et la variable ANNEE en position 15, puis de déclarer une donnée auto-générée mappant vers le DTM et ayant pour valeur :

```
$SUB (0,2,$DATE2) + $ANNEE + $MOISJOUR.
```

8. CONSTANTES ET EXPRESSIONS

TX2 offre un évaluateur d'expressions, qui permet de générer des données EDI ou des données applicatives. Cet évaluateur permet de manipuler des constantes, des variables (voir plus haut), et des fonctions pré-définies.

Dans sa version actuelle, cet évaluateur d'expressions ne permet de manipuler que des chaînes de caractères.

8.1. Constantes :

Les constantes sont des chaînes apparaissant entre **""**. Par exemple **"HB TECHNOLOGIES"** ou **"1000"**.

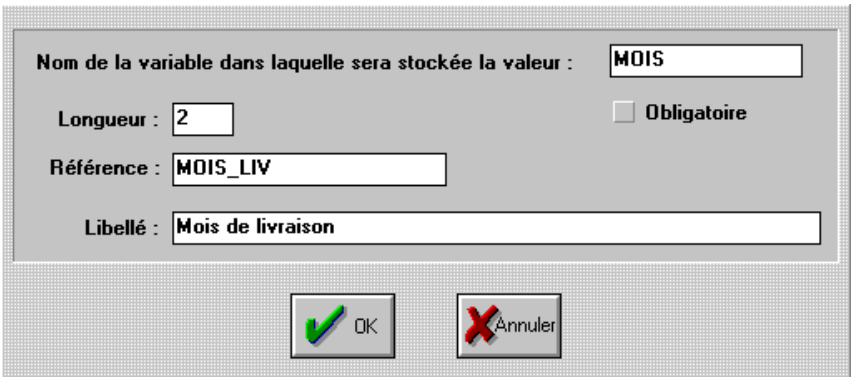
8.2. L'opérateur + :

L'opérateur + permet de concaténer 2 constantes ou expressions. Par exemple :

"HB " + "Technologies "
donne **"HB Technologies "**.
"LE " + $DATE donne **"LE 960201"**.

8.3. Les variables :

Les variables définies par l'utilisateur doivent être préfixées par le signe $ lorsque leur valeur est utilisée dans une expression. Par exemple si on a défini la variable MOIS comme indiqué dans l'écran :



on pourra la référencer comme dans l'exemple suivant :

"A LIVRER LE MOIS " + $MOIS

8.4. Les fonctions pré-définies :

Dates :

$DATE : Renvoie la date système sous la forme AAMMJJ.

$DATE2 : Renvoie la date système sous la forme CCAAMMJJ.

$TIME : Renvoie l'heure système sous la forme HHMM.

$TIMES : Renvoie l'heure système sous la forme HHMMSS

$CC (<Année>) : FONCTION AN 2000. Renvoie le siècle à partir de l'année sur 2 positions, selon la logique suivante : Si année >= 50, alors le siècle vaut 19, sinon le siècle vaut 20.

Exemples :

Expression	**Résultat**
`$CC ("98")`	`19`
`$CC($SUB(0,2,$DTLIV))`	`20` (si $DTLIV="021024")

Compteurs :

$I : Renvoie la valeur du compteur I

$I+=J : Renvoie la valeur du compteur I après l'avoir incrémenté de J.
Exemple : $3+=1 : incrément du compteur 3 d'une valeur égale à 1.

$I=+J : Renvoie la valeur du compteur I puis l'incrémente de J.

$I-=J : Renvoie la valeur du compteur I après l'avoir décrémenté de J.

$I=-J : Renvoie la valeur du compteur I puis le décrémente de J.

$1 .. $32 : 32 compteurs sauvegardés entre chaque session.

$_MESS1 .. $_MESS128 : 128 compteurs initialisés à zéro à chaque nouveau message.

$_SESS1 .. $_SESS128 : 128 compteurs initialisés à zéro à chaque nouvelle session de traduction.

Arithmétique :

$+(EXPR1 , EXPR2 , <NBDEC>) : Additionne EXPR1 et EXPR2
$-(EXPR1 , EXPR2 , <NBDEC>) : Soustrait EXPR1 et EXPR2
$*(EXPR1 , EXPR2 , <NBDEC>) : Multiplie EXPR1 et EXPR2
$/(EXPR1 , EXPR2 , <NBDEC>) : Divise EXPR1 et EXPR2

Toutes ces opérations sont effectuées en double précision (représentation flottante sur 64 bits).
Le nombre de décimales est opérationnel, s'il n'est pas précisé, le résultat obtenu est dépaddé de ses zéros.

Exemple : $+(1 ,2) Retourne « 3 »
$+(1 ,2, 3) Retourne « 3.000 »

$//(EXPR1 , EXPR2) Division entière de EXPR1 par EXPR2

Opérations sur les chaînes de caractères :

+ Concaténation de chaînes de caractères.

Exemple : **« XXX » + « YYY »** Retourne « XXXYYY »

$SUB ([Début],[Longueur],[Expression]) : Retourne la sous-chaîne de la chaîne ***expression*** en commençant à la position ***Début*** sur ***Longueur*** positions. Attention, le premier caractère de la chaîne est en position 0.

Exemples :

`$SUB(0,2,"HBTECHNO")` donne **`HB`**
`$SUB(2,6,"HBTECHNO")` donne **`TECHNO`**

$SUBST ([Expr],[Caractère]
$PAD ([Longueur],[L/R],[Caractère],[Expression]) : Padde la chaîne ***expression*** à gauche (***L***) ou à droite (***R***) sur ***Longueur*** positions en utilisant le ***Caractère***.

Exemples :

`$PAD(8,L,"0","120")` donne **`00000120`**
`$PAD(10,R," ","DH") + "X"` donne **`DH X`**

$PADNUM([Longueur],[L/R],[Caractère],[Expression])
Idem mais avec une expression numérique, et tient compte du signe de celle-ci.

Exemple :

$PADNUM (5 , L, " " , "-5") donne **""- 5"**

$UNPAD([Longueur],[L/R],[Caractère],[Expression]) : Dépadde la chaîne de caractères EXPR du caractère CHAR, le résultat obtenu ayant une longueur minimale de LONG.

Exemple :

$UNPAD (3,L,' '," 3333") donne **"**3333**"**
$UNPAD (3,L,' '," 1") donne **"** 1**"**

Divers :

$NULL Retourne la chaîne de caractères vide

$IF ([Expr1],=,[Expr2],[Expr3],[Expr4]) : Retourne la chaîne ***expr3*** si ***expr1*** est égale à ***expr2***, et retourne ***expr4*** si ***expr1*** n'est pas égale à **expr2**.
Question : Seulement le signe = ?
Exemples : si nom vaut "HB" et prénom vaut "DANIEL" :

`$IF($NOM,=,"HB","NOM VAUT HB","PERDU")` donne **`NOM VAUT HB`**
`$IF($PRENOM,=,"JEAN","OK","")` donne un résultat 'Chaîne vide'.

$SET(VAR,EXPR)
Affecte la valeur EXPR à la variable VAR, puis retourne cette valeur.

Exemple : **$SET($X,"VALEUR")**
La variable $X prend la valeur **"**VALEUR**"**.

$OUT(EXPR)
En réception seulement, affecte la valeur de EXPR au nom du fichier de sortie qui contiendra les documents traduits en format interne, l'extension sera toujours .RCV

Exemple : **$OUT("XXX")**
Le nom du fichier de sortie sera XXX.RCV

Exemple d'utilisation : on souhaite dissocier parmi des messages ORDERS reçus les ORDERS originales (donnée BGM 1225 = 9) des autres commandes. Dans ce cas, on stockera la valeur de la donnée BGM 1225 dans la variable `MESFONC` et on utilisera l'expression :

`$OUT($IF($MESFONC,=,"9","ORIGINAL","AUTRES"))`

Les commandes originales seront routées vers `ORIGINAL.RCV`, les autres vers `AUTRES.RCV`.

$IN
Retourne le nom du fichier d'entrée (par exemple DELINS.INH en émission et G-GEIS.R en réception).

$ARG1 .. $ARG16 Retourne la valeur du énième paramètre externe de la commande de traduction (identifié par la chaîne -ARGn) passé sur la ligne de commande (-2 en réception et -1 en émission).

Exemples :

```
TX2RCV M-ATLAS.R -2 -ARG1 Argument1 -ARG2 Argument2  -Oq
TX2SND ORDERS.INH ORDERS -1 -Rb -ARG1 Argument1 - ARG2 Argument2
```

Les variables $**ARG1** et $**ARG2** valent respectivement "Argument1" et "Argument2" les autres sont vides.

$NBMESS
En réception seulement, retourne le nombre de messages dans la session de traduction.

Gestion de piles de données :

TX2 permet de gérer des piles de données. Ceci est utile en réception, en général lorque l'on souhaite référencer dans un enregistrement mappant sur un segment de niveau n une ou plusieurs données mappant sur des niveaux n+i, ces segments étant mappés par ailleurs.

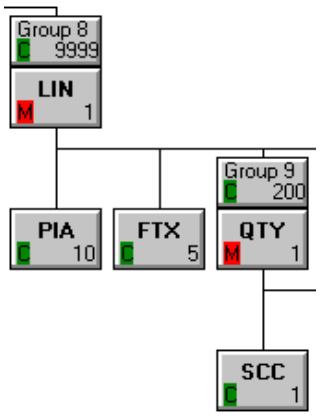
La mise en œuvre des fonctions de gestion de piles se fait en utilisant les fonctions pré-définies **$STACK** et **$WRITE** :

$STACK("Identifiant de la pile",EXPR)
Exemple : `$STACK("SCC1",$SUB(0,1,$TYPPROG) + "001")`

$WRITE("Identifiant de la pile")
Exemple : `$WRITE("SCC1")`

Exemple d'utilisation :

Considérons le diagramme de branchement suivant :



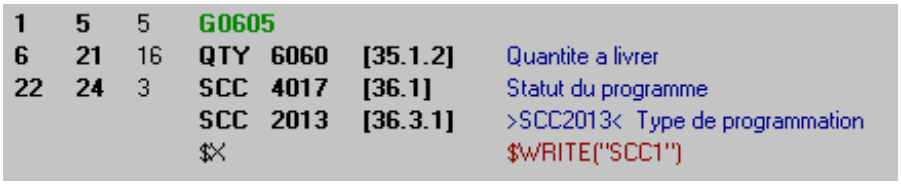
On souhaite générer 2 type d'enregistrements avec les caractéristiques suivantes :

L'enregistrement **A0900** mappe sur les segments LIN et PIA (1 occurrence par LIN), et l'enregistrement **G0605** mappe sur le groupe SCC-QTY. Jusqu'ici, il n'y a pas de problème. Mais de plus, il faut écrire dans l'enregistrement A0900 une donnée du segment SCC, qui doit correspondre à la première répétition du groupe SCC pour un qualifiant de QTY égal à 113. Ceci ne peut pas se faire en mappant simplement la donnée du SCC dans le segment A0900, car dans ce cas TX2 ne verra plus les répétitions du SCC. Il faut utiliser la fonction **$STACK** dans le A0900 et la fonction **$WRITE** dans le G0605 :

Enregistrement A0900 :

```
                    LIN   1229   [19.2]     >LIN1229<  action de mise a jour, code
6     40    35      LIN   7140   [19.3.1]   Numero d'article
41    75    35      PIA   7140   [20.2.1]   Numero d'article du fabricant
                    --Type de programmation
76    78    3       $STACK("SCC1",$TYPPROG)
79    81    3       $LIN1229
```

Enregistrement G0605 :


1 5 5 G0605
6 21 16 QTY 6060 [35.1.2] Quantite a livrer
22 24 3 SCC 4017 [36.1] Statut du programme
SCC 2013 [36.3.1] >SCC2013< Type de programmation
$X $WRITE("SCC1")

L'instruction **$STACK("SCC1",$typprog)** dans l'enregistrement A0900 indique au système qu'à cet endroit l'expression **$typprog** devra être générée à partir de la pile SCC1, lorsque TX2 rencontrera un ordre d'écriture de cette pile. Dans cet exemple, cet ordre d'écriture (ou dépilage) est programmé dans l'enregistrement G0605 : c'est l'ordre **$WRITE ("SCC1")**, qui doit être enregistré dans une cellule de type "*calcul*".

Il est possible de gérer plusieurs piles dans un même format interne, et pour chaque pile il est possible de gérer plusieurs expressions ou variables dans plusieurs instructions $STACK.

L'utilisation à bon escient du couple `$STACK / $WRITE`, par la capacité qu'il procure à évaluer des expressions "rétroactivement", permet de résoudre la plupart des cas complexes de mises en correspondance de structures a priori incompatibles.

8.5. L'instruction SWITCH (instructions conditionnelles) :

Il est possible d'enregistrer des groupes d'instructions conditionnelles, en utilisant l'instruction SWITCH. Ceci se fait sous forme de cellule Texte libre.

Voici un exemple de groupe d'instructions SWITCH :

```
DATA (3)  "QN" MAPTO SKIP;
DATA (35) "VN" MAPTO SKIP;

SWITCH($QN) {
   CASE ("DP") :
      CONSTANT DATA (VAL = $vn ) MAPTO 3.1 ;
   CASE ("SU") :
      CONSTANT DATA (VAL = $vn ) MAPTO UNB.3.1 ;
   CASE ("BY") :
     CONSTANT DATA (VAL = $vn ) MAPTO UNB.2.1 ;
   ELSE :
     CONSTANT DATA (VAL = "ZZ" ) MAPTO UNB.2.1 ;
};
```

Cet exemple illustre bien la structure d'un bloc **SWITCH**, qui permet de conditionner des instructions sur la valeur d'une variable (ici la variable **QN**). Dans cet exemple, on alimente tel ou tel segment selon la valeur d'un qualifiant. Ainsi si **$QN** vaut **SU**, alors on alimente l'adresse de l'émetteur avec le contenu de **VN**.

9. SAUVEGARDER.

Lorsque vous définissez un format interne, sauvegardez le régulièrement en utilisant l'option de menu Fichier - Enregistrer.

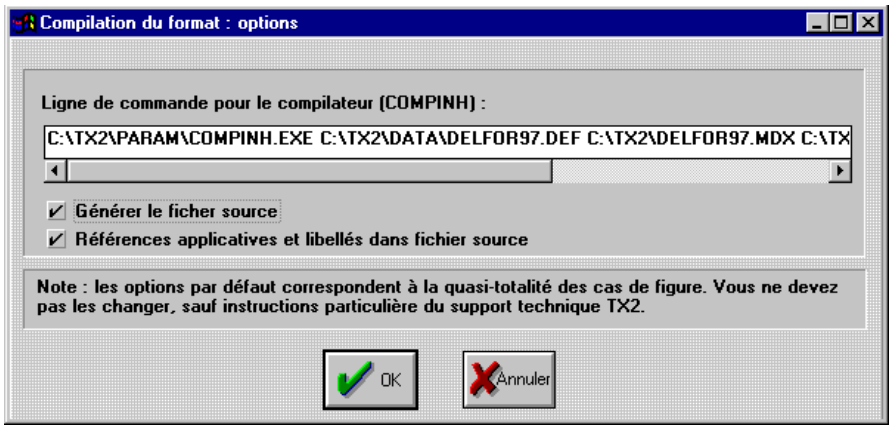
Le système crée alors 3 fichiers : un fichier **\TX2\DATA\<Code format>.REW** (dans lequel sont stockées les informations saisies, sous la forme base de données), un fichier **\TX2\DATA\<Code format>.SEQ** (dans lequel sont stockées les informations liées à la structure générale - numéros de

séquence, structures, répétitions) et un fichier **\TX2\DATA\<Code format>.DEF**, qui est le fichier source de paramètrage, celui-là même qui sera compilé. Voici un exemple de fichier **DEF** :

```
FORMAT=FIXED;
PAD=DEFAULT." ";

/* Constants définitions */

DATASET {
  DATA "DATE_EXP" (0) MAPTO 4.1.2  (WITH 4.1.1="200") ;   -- DTM.2380
  DATA "DATE_LIV" (0) MAPTO 4.1.2  (WITH 4.1.1="2") ;   -- DTM.2380
  DATA "CODE_EXPE" (0) MAPTO 39.2.1  (WITH 39.1="PW") ;   -- NAD.3039
  DATA "NOM_EXPE" (0) MAPTO 39.3.1  (WITH 39.1="PW") ;   -- NAD.3124
  DATA "OBSERV" (0) MAPTO 66.4.1 ;     -- FTX.4440
  DATA "COLIS" (0) MAPTO 58.2.1 ;      -- GID.7224
  DATA "QTE-COMP" (0) MAPTO 9.1.2  (WITH 9.1.1="6") ;   -- CNT.6066
  DATA "POIDS-BR" (0) MAPTO 9.1.2  (WITH 9.1.1="7") ;   -- CNT.6066
  DATA "REF-EXT" (0) MAPTO 14.1.2  (WITH 14.1.1="ON") ;   -- RFF.1154
  DATA "UN-QTECP" (0) MAPTO 9.1.3 (WITH 9.1.1="6") ;   -- CNT.6411
       }

/* Records définitions */

SET {
     RECORD {
       LABEL = "R";
       DATA (0) MAPTO 1.1.1 ;               -- BGM.1001
   TARGET DATA (4,PICTURE=DATE."MMDD"."YYYYMMDD") MAPTO "DATE_EXP" ;
   TARGET DATA (4,PICTURE=DATE."MMDD"."YYYYMMDD") MAPTO "DATE_LIV" ;
       DATA (2) MAPTO SKIP;                                -- 15  -> 16
   TARGET DATA (2,PICTURE=DATE."YY"."CCYYMMDD") MAPTO "DATE_EXP" ;
       DATA (1) MAPTO SKIP;                 --          -- 20  -> 20
       TEXT (LENGTH=6,"000000");
   TARGET DATA (6) MAPTO "CODE_EXPE"  (CODE = TIERS);
   TARGET DATA (5) MAPTO "CPOS-EXP" ;
       TEXT (LENGTH=1,"2");
   TARGET DATA (24) MAPTO "OBSERV" ;
       TEXT (LENGTH=2,"  ");
   TARGET DATA (4,PICTURE=NU.0) MAPTO "COLIS" ;
   TARGET DATA (6,PICTURE=NU.0) MAPTO "POIDS-BR" ;
   TARGET DATA (6,PICTURE=NU.0) MAPTO "QTE-COMP" ;
       DATA (12) MAPTO SKIP;                               -- 181 -> 192
   TARGET DATA (8) MAPTO "REF-EXT" ;
   TARGET DATA (2) MAPTO "UN-QTECP" ;
   TARGET DATA (8) MAPTO "CODE_DEST" ;
       DATA (46) MAPTO SKIP;                        -- 211 -> 256
     }
}
```

10. COMPILER.

Pour pouvoir être exploité par le traducteur, un format interne doit avoir été compilé. Pour ce faire, sélectionnez l'option **Fichier - compiler** :


Compilation du format : options
Ligne de commande pour le compilateur (COMPINH) :
C:\TX2\PARAM\COMPINH.EXE C:\TX2\DATA\DELFOR97.DEF C:\TX2\DELFOR97.MDX C:\TX
Générer le ficher source
Références applicatives et libellés dans fichier source
Note : les options par défaut correspondent à la quasi-totalité des cas de figure. Vous ne devez pas les changer, sauf instructions particulière du support technique TX2.
OK
Annuler

Si vous souhaitez compiler un fichier .DEF conçu par éditeur sans le ré-écraser, dévalidez le bouton "**Générer le fichier source**".
Le bouton "**Références applicatives**..." permet de stocker dans le fichier compilé les libellés des données, qui seront utiles pour la documentation du format.

Si la compilation s'est bien passée, le système crée un fichier **`\TX2\<Code format>.FDX`**. C'est ce fichier qui sera utilisé par le traducteur.Si la compilation s'est mal passée, le fichier .FDX n'est pas créé. En règle générale, les erreurs de compilation correspondent au fait que des données ou segments obligatoires n'ont pas été référencés dans un format Emission ou Emission+Réception, ou à des erreurs dans des expressions de constantes ou calculs.

11. TABLES DE TRANSCODIFICATIONS.

Les tables de transcodifications sont créées et maintenues au travers d'un utilitaire. Pour accéder à cet utilitaire, cliquer sur l'icône Transcos dans le groupe TX2 - Station EDI.

Pour plus de détails, consulter le manuel d'utilisation de l'utilitaire WTRANSCOS.

12. PARTICULARITES DES FORMATS EDI->EDI.

Pour traiter un format EDI->EDI, TX2 effectue de manière automatique une mise en format plat étendu du message EDI entrant, puis considère ce fichier plat comme un fichier format interne classique.

Il y a donc 2 étapes pour définir un format EDI->EDI :

1. Définir le subset du message entrant (Atelier de conception), et générer le format standard EDI->EDI (Atelier de conception). Définir également le subset du message sortant le cas échéant.

2. Dans le mapper, créer un format interne, et donner comme nom le nom du format généré dans l'étape 1. Ce format défini automatiquement les enregistrements du fichier plat, en nommant toutes les variables. Il suffit alors d'alimenter les données du message sortant en utilisant des instructions Auto-générée. On peut également mapper directement, mais **en aucun cas il ne faut changer les tailles des enregistrements et des données.**

13. PARTICULARITES DU STANDARD GENCOD.

Les règles générales définies dans les pages qui précèdent sont applicables au standard GENCOD.

13.1. Gestion des rubriques 177.

Les rubriques 177 peuvent apparaître n'importe où dans le message. Il est possible de mapper une rubrique 177 en la référençant par rapport à la rubrique immédiatement précédente.

Par exemple, si on s'attend à recevoir la séquence 290 141 177 177 290 141, on référencera les rubriques 177 de cette séquence en déclarant des mappings 141.TXT1 et 141.TXT2. Ceci indique au système que ces rubriques sont dépendantes du 141.

13.2 Segments de service.

Le traducteur passe par un format intermédiaire "pseudo-EDIFACT". Pour cette raison, le mappeur génère dans le fichier .DEF les lignes suivantes :

```
DATASET {
        CONSTANT DATA (VAL=$DATE) MAPTO UNB.4.1;
        CONSTANT DATA (VAL=$TIME) MAPTO UNB.4.2;
        DATA (0) MAPTO 1.1;
        DATA (0) MAPTO 2.1;
        DATA (0) MAPTO 2.2;
        }
```

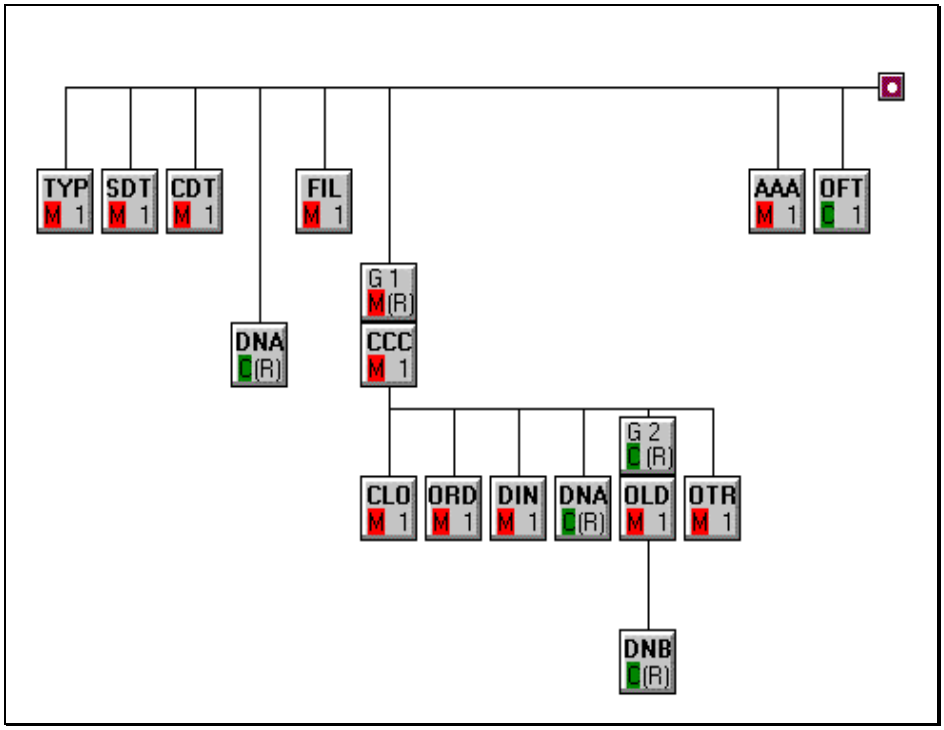
Cette gestion est transparente à l'utilisateur.

14. PARTICULARITES DU STANDARD TRADACOM.

La particularité du standard TRADACOM réside dans le fait qu'un document se trouve représenté par plusieurs messages au sens EDIFACT. Un message se trouve encadré par les segments MHD et MTR, et un interchange (représentant souvent 1 seul document) se trouve encadré par les segments STX et END.
Afin de pouvoir modéliser une telle structure, il faut créer une structure de message unique dans l'atelier de conception. Par exemple, le message Commande modélisé par :

```
ORDHDR          1          de TYP...à FIL
ORDERS          R          de CLO...à OTR
ORDTLR          1          de OFT...à OFT
```

sera décrit dans l'atelier de conception comme suit :



En réalité, les segments CCC et AAA représentent les segments MHD pour les messages ORDERS et ORDTLR. De plus, on ne fait pas apparaître dans la structure les segments MTR.

Cette règle doit toujours être respectée : il faut définir un message composite, en remplaçant les segments MHD par des segments dont le nom est ***impérativement sous la forme XXX***. Bien entendu, ces segments doivent exister dans le répertoire des segments.

la forme XXX. Bien entendu, ces segments doivent exister dans le répertoire des segments.

Cas d'un paramétrage Format interne vers format EDI :

Dans ce cas, le mapping doit être réalisé comme pour un message EDIFACT. Les données des segments XXX doivent être alimentées comme constantes (par exemple ORDTLR, 8).

Cas d'un paramétrage Format EDI vers format Interne :

Dans ce cas, le mapping doit être réalisé comme pour un message EDIFACT, mais il faut en plus mettre à jour le fichier \TX2\MULTISTD.INI, afin d'indiquer au traducteur qu'il doit transcoder les MHD entrants en AAA ou CCC, selon le type de message. Dans notre cas, le fichier MULTISTD.INI contiendra :

```
[TRADACOM]
;Syntaxe : UNH=Code message,<Transco MHD>,<Transco MTR>
UNH=ORDHDR:6,UNH,MTR
UNH=ORDERS:8,AAA,MTR
UNH=ORDTLR:4,BBB,UNT
```

Reconnaissance d'un message TRADACOM :

Dans l'atelier de conception, un message Tradacom doit être créé ou importé en spécifiant les données UNH suivantes :

<Type message>:<Version>:TRA:HB:TRADAC

Par exemple dans notre cas : **ORDHDR:6:TRA:HB:TRADAC**

15. PARTICULARITES DU STANDARD VDA.

Le standard VDA est très proche du standard GENCOD, en ce sens qu'il s'agit d'un format de type fixe, sans identification formelle de rubrique.

La contrainte additionnelle est qu'il faut maintenir un fichier **\TX2\VDA.INI**, qui doit permettre au traducteur TX2 d'identifier les messages VDA entrant.

Voici la structure du fichier VDA.INI :

[RECEIVE]	***Exemple***
DOC=<Type doc>	*4916*
TAG=<Tag début>,<Tag fin>	*661,669*
NUMI=<Tag No interchange>,<Pos début>,<Longueur>	*661,29,5*
NUMM=<Tag No message>,<Pos début>,<Longueur>	*661,29,5*
DEST=<Tag id dest>,<Pos début>,<Longueur>	*661,6,9*
EMET=<Tag id emet>,<Pos début>,<Longueur>	*661,15,9*
UNHDEF=<Id. UNH>	*V4916,1,01,VD,VDA*
FINDOC=TRUE	
DOC=...	
TAG=...	
...	
FINDOC=TRUE	

La page suivante contient un exemple de fichier VDA.INI.

```
[RECEIVE]
DOC=4916
TAG=661,669
NUMI=661,29,5
NUMM=661,29,5
DEST=661,6,9
EMET=661,15,9
UNHDEF=V4916,1,01,VD,VDA
FINDOC=TRUE
```

Dans cet exemple, le message de type 4916 débutera par un enregistrement 661 et se terminera par un enregistrement 669. Le numéro d'échange et le numéro de message seront trouvés dans l'enregistrement 661, en position 29 sur une longueur de 5, le destinataire sera trouvé dans l'enregistrement 661, en position 6 sur une longueur de 9, l'émetteur sera trouvé dans l'enregistrement 661, en position 15 sur une longueur de 9, et l'identification "TX2" du message est V4916:1:01:VD:VDA.

Note : Les messages VDA doivent être identifiés dans l'atelier de conception sous la forme :

VXXXX,1,01,VD,VDA

où **XXXX** est le code du type de message VDA (par exemple 4916).

ARCHITECTURE GENERALE DE L'ENVIRONNEMENT DE DEVELOPPEMENT TX2

WMESS : Atelier de conception de messages et subsets
WINH : Mapper
COMPINH : Compilation et contrôle de cohérence des messages et mappings
WTX2 : Run-time de traduction TX2

Annexe

Gestion des tables de références INHREF et MESSREF

Les tables de référence **\TX2\INHREF.RDX**, **\TX2\MESSREF.RDX** et **\TX2\SUBSREF.RDX** sont des fichiers binaires dans lesquels sont stockés les liens ***Format <-> Message/Subset <-> Identification UNH***.

Ces tables sont mises à jour automatiquement par l'Atelier de conception et le Mapper lors de l'enregistrement des messages et lors de l'enregistrement et la compilation des formats internes.

Par ailleurs, ces tables sont exploitées par le run-time du traducteur TX2, en particulier pour retrouver quels formats internes utiliser pour les documents reçus (identifiés par l'UNH, le MHD, etc... selon le standard). Lorsque l'environnement de développement et le traducteur sont installés sur le même ordinateur, la cohérence de ces tables est automatiquement assurée.

Par contre, lorsqu'on a besoin de transférer des paramétrages d'un poste à un autre, ou lorsqu'on souhaite gérer plusieurs configurations utilisateurs, il est nécessaire de maintenir correctement les tables de référence. *En particulier, la procédure d'installation de la station TX2 permet de mettre à jour automatiquement les tables de référence, à partir de fichiers de commande à créer sur la disquette d'installation (Cf documentation technique, installation de paramètrages).*

Cependant il existe un moyen plus souple de gérer les tables de paramètrages, grâce à l'utilitaire **BPT.EXE** qui permet de gérer ces tables sous forme de fichiers Texte, en permettant de convertir ces tables en format texte et inversement (***Win 95,98 et NT seulement***).

L'exécutable **BPT.EXE** est installé dans le répertoire **\TX2**. Au lancement, le répertoire courant doit être **\TX2**.

La syntaxe d'utilisation est :

```
BPT.EXE      RDX2TXT      MESS
             TXT2RDX      SUBS
                          INH
```

Le premier paramètre indique le sens de conversion : **RDX2TXT** pour une conversion de binaire vers texte, **TXT2RDX** pour une conversion de texte vers binaire. Les fichiers texte portent le même nom que les fichiers binaires **RDX**, et leur extension est **TXT**.

Le deuxième paramètre indique le type de la table traitée : **MESS** pour la table des messages (MESSREF.RDX), **SUBS** pour la table des subsets (SUBSREF.RDX), et **INH** pour la table des formats (INHREF.RDX).

Par exemple :

```
BPT.EXE RDX2TXT INH      : de INHREF.RDX en INHREF.TXT
BPT.EXE TXT2RDX MESS     :de MESSREF.TXT en MESSREF.RDX
```